

A NEWTON-SCHULZ VARIANT FOR IMPROVING THE INITIAL CONVERGENCE IN MATRIX SIGN COMPUTATION

JIE CHEN* AND EDMOND CHOW†

Abstract. The Newton-Schulz iteration is a quadratically convergent, inversion-free method for computing the sign function of a matrix. It is advantageous over other methods for high-performance computing because it is rich in matrix-matrix multiplications. In this paper we propose a variant that improves the initially slow convergence of the iteration for the Hermitian case. The main idea is to design a fixed-point mapping with steeper derivatives at the origin in order to accelerate the convergence of the eigenvalues with small magnitudes. In general, the number of iterations is reduced by half compared with standard Newton-Schulz; and, with proper shifts, the number can be further reduced. We demonstrate numerical calculations with matrices of size up to the order of 10^4 – 10^5 on medium-sized computing clusters and also apply the algorithm to electronic-structure calculations.

Key words. Matrix sign function, Newton-Schulz iteration, electronic-structure calculation

AMS subject classifications. 65F60

1. Introduction. We are interested in numerically computing the matrix sign function

$$S = \text{sign}(A)$$

for a matrix $A \in \mathbb{C}^{n \times n}$ with no eigenvalues lying on the imaginary axis. The scalar sign function $\text{sign}(z)$ takes value $+1$ when $\Re(z) > 0$ and -1 when $\Re(z) < 0$. Although much of the existing theory and methods cover the general case where the eigenvalues of A are complex, we focus our approach in this paper on the Hermitian case such that the eigenvalues are all real. Hence, a simplified definition of the matrix sign function for this case is

$$\text{sign}(A) = U \cdot \text{diag}(\text{sign}(\lambda_1), \dots, \text{sign}(\lambda_n)) \cdot U^*,$$

where $U^*AU = \text{diag}(\lambda_1, \dots, \lambda_n)$ is a diagonalization of A . The Hermitian case appears in several real-life applications, such as lattice quantum chromodynamics [19, 24, 8] and electronic-structure calculations [23, 21]. For the latter application, the exact density matrix ρ of a molecular system with Fermi level μ is the Heaviside function of $\mu I - H$, where H is an approximation to the Hamiltonian. Thus, one can compute ρ as $\frac{1}{2}[\text{sign}(\mu I - H) + I]$. More details of this application are given in Section 8.

Several numerical methods exist for computing $\text{sign}(A)$. Here, we discuss the methods for the general case and mention the special treatment of the Hermitian case, if any. Whenever applicable, we discuss the potential for parallelization of a method.

The Schur method [9, Section 5.2] is considered a direct method. The method first computes a Schur decomposition $A = QTQ^*$ and then applies the Parlett recurrence on T , yielding $R = \text{sign}(T)$. Hence, $\text{sign}(A)$ is formed as QRQ^* . When A is

*Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439 (jiechen@mcs.anl.gov). Work supported by the U.S. Department of Energy under Contract DE-AC02-06CH11357.

†School of Computational Science and Engineering, College of Computing, Georgia Institute of Technology, Atlanta, GA 30332 (echow@cc.gatech.edu). Work supported by NSF under grant ACI-1147843.

Hermitian, the Schur decomposition is a diagonalization; hence, T is diagonal, and the Parlett recurrence is not needed. Whereas the Schur decomposition and the eigenvalue decomposition have been demonstrated to have good performance on multicore machines by using packages such as ScaLAPACK [1], the efficient parallelization of the decompositions on distributed memory machines is much more difficult. This situation limits parallel scalability.

Two iterative methods that are the most relevant to the present paper are Newton's method and the Newton-Schulz method [11, 9]. Newton's method starts with $X_0 = A$ and iterates through

$$X_{k+1} = \frac{1}{2}(X_k + X_k^{-1}), \quad k = 0, 1, \dots$$

The iterates X_k always converge to S (see [9, Theorem 5.6]). It is straightforward to write $X_{k+1} - S = \frac{1}{2}X_k^{-1}(X_k - S)^2$; hence,

$$\lim_{X_k \rightarrow S} \frac{\|X_{k+1} - S\|_2}{\|X_k - S\|_2^2} \leq \frac{1}{2}\|S^{-1}\|_2 = \frac{1}{2}.$$

This means that Newton's method converges quadratically.

On the other hand, the Newton-Schulz method starts with $X_0 = A$ and iterates through

$$X_{k+1} = \frac{1}{2}X_k(3I - X_k^2), \quad k = 0, 1, \dots \quad (1.1)$$

We have the following convergence result.

THEOREM 1.1. *The Newton-Schulz iteration is quadratically convergent when $\|I - A^2\|_2 < 1$.*

Proof. We write $I - X_{k+1}^2 = I - \frac{1}{4}[I - (I - X_k^2)][2I + (I - X_k^2)]^2$. Hence, denoting by $G_k = I - X_k^2$, we obtain $G_{k+1} = \frac{3}{4}G_k^2 + \frac{1}{4}G_k^3$. If initially $\|G_0\|_2 < 1$, by induction one sees that $\|G_{k+1}\|_2 < \|G_0\|_2^{2^k}$. Then, $G_k \rightarrow 0$. By factorizing G_k as $(S - X_k)(S + X_k)$ and noting that $S + X_k$ is always nonsingular, we conclude that $X_k \rightarrow S$. We now rewrite $\|G_{k+1}\|_2 < \|G_0\|_2^{2^k}$ as

$$\|X_{k+1} - S\|_2 < \|(X_{k+1} + S)^{-1}\|_2 \cdot \|I - A^2\|_2^{2^k}$$

to show the quadratic rate of convergence. \square

In the Hermitian case, the condition $\|I - A^2\|_2 < 1$ that ensures convergence is equivalent to requiring that the spectral radius $\rho(A)$ is less than $\sqrt{2}$. Therefore, for the method to be applicable, one must first compute the spectral radius and prescale A . This computation can be done by using the Lanczos algorithm for large matrices and thus is inexpensive. An advantage of the Newton-Schulz method, compared with Newton's method, is that the former is rich in matrix-matrix multiplications. Hence, the Newton-Schulz iteration is easier to parallelize and is expected to scale much better than when matrix inverses are required, as in Newton's iteration. We note that in electronic-structure calculations, the McWeeny purification method [17, 14] is equivalent to the Newton-Schulz method.

The Newton-Schulz method is an instance of the Padé family of iterations $X_{k+1} = f_{\ell,m}(X_k)$, where [10]

$$f_{\ell,m}(x) = \frac{xp_{\ell}(1 - x^2)}{q_m(1 - x^2)}$$

and $p_\ell(x)/q_m(x)$ is the $[\ell/m]$ Padé approximant of the function $(1-x)^{-1/2}$. (Take $\ell = 1$ and $m = 0$ for Newton-Schulz.) The Padé iteration for matrices is thus

$$X_{k+1} = X_k p_\ell(1 - X_k^2) q_m(1 - X_k^2)^{-1}.$$

For convergence of the iteration and other interesting properties, see [9, Section 5.4]. Here, we consider the computational aspects. When $m = 0$, $f_{\ell,m}$ is a polynomial of degree $2\ell + 1$. Evaluating this polynomial requires $\ell + 1$ matrix-matrix multiplications. Whether it is best to use $\ell = 1$ (that is, Newton-Schulz) or a higher value depends on the speed of convergence; an explicit rule is unclear. On the other hand, when $m > 1$, matrix inversions are necessary. One can directly form $q_m(1 - X_k^2)$, followed by an inversion (which is the only one), or use the continued-fraction representation of the Padé approximant, which requires $2m - 1$ inversions. On state-of-the-art parallel computer architectures, matrix inversions scale less satisfactorily than do matrix multiplications.

The Padé approximation belongs to a broader category of rational approximations. Coincidentally, the best uniform approximation of the sign function on a pair of symmetric but disjoint intervals can be expressed as a rational function (see, e.g., [9, Theorem 5.15]). Thus, when A is Hermitian, $\text{sign}(A)$ is approximated by a rational function of A , the calculation of which requires matrix inversion as well.

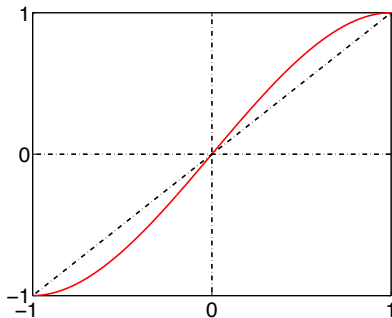
The Newton-Schulz method is appealing in a large-scale setting because it is inversion free. We focus on improving the method in this paper, by noting that a drawback of the method is that it requires a large number of iterations before quadratic convergence is seen. In Section 2 we investigate this phenomenon in depth by interpreting the iteration as a fixed-point mapping. The main discovery is that the initial convergence for the eigenvalues closest to the origin is slow and that the derivative of the mapping at the origin is of vital importance. Thus, in Section 3 we derive a new mapping that maximizes the derivative at the origin so as to improve the initial convergence. Detailed analysis follows (see Sections 4 and 5), with a particular result suggesting that the number of iterations for the new mapping is reduced by one half compared with the Newton-Schulz mapping. Practical implementation of the new method is discussed in Section 6. At the end of the paper we demonstrate large-scale calculations, including in parallel (see Section 7), and the application of computing the density matrix in electronic-structure theory (see Section 8).

Terminology and notation. To avoid wordiness, we use the term “smallest/largest magnitude eigenvalue” of a matrix A to indicate the smallest/largest element of the set $\{|\lambda(A)|\}$, where $\lambda(A)$ denotes the eigenvalues of A . These two elements are denoted as $\lambda_{|\min|}(A)$ and $\lambda_{|\max|}(A)$, respectively. They are not necessarily the eigenvalues of A ; only the magnitude matters.

2. Standard Newton-Schulz. From here on, we refer to the Newton-Schulz iteration (1.1) as “standard” Newton-Schulz. We use f to denote the Newton-Schulz mapping

$$f(x) = \frac{1}{2}x(3 - x^2) \tag{2.1}$$

so that the Newton-Schulz iteration reads $X_{k+1} = f(X_k)$. Since A is Hermitian, the convergence of Newton-Schulz is completely characterized by the properties of f on the real line. Since we can always scale the matrix, we consider only the interval $x \in [-1, 1]$, and we assume that $\rho(A) = 1$. Then, Newton-Schulz is always convergent according to Theorem 1.1. Figure 2.1 plots f .

FIG. 2.1. Mapping f .

Because f is odd, we further restrict our attention to the interval $[0, 1]$. The mapping f on $[0, 1]$ is monotonically increasing and admits $x < f(x)$, except when $x = 0$ or 1 . Hence, one intuitive explanation of why the iteration $X_{k+1} = f(X_k)$ converges to the sign of A is that f pushes all the positive eigenvalues of A toward 1 in a monotonic manner (and similarly pushes the negative eigenvalues toward -1). Among all these eigenvalues, the one that converges the most slowly is the eigenvalue closest to the origin. Let this eigenvalue be x_0 , and without loss of generality assume that $x_0 > 0$. Then, the initial iteration reduces the condition number from $1/x_0$ to $1/f(x_0)$. When A is ill-conditioned (i.e., $x_0 \approx 0$), the rate of reduction is $f(x_0)/x_0 \approx f'(0)$. Because of the significance of $f'(0)$, one naturally asks what is the optimal mapping in the sense that the derivative at the origin is maximal. It turns out that the optimal mapping is the one defined in (2.1), as the following result states.

THEOREM 2.1. *Let P be the set of cubic and odd polynomials that are monotonically increasing on the interval $[0, 1]$ and that maps this interval to itself. Then,*

$$f = \arg \max_{g \in P} g'(0) \quad \text{with} \quad f'(0) = \frac{3}{2},$$

where f is defined in (2.1).

Proof. The polynomial must pass the origin because it is odd. It also must pass the point $(1, 1)$ because it is monotonically increasing. Then, all such cubic polynomials must have the form $g = ax + (1 - a)x^3$. The monotonic increase implies that $a \leq \frac{3}{2}$. Thus, $g'(0) = a$ is maximized when $a = \frac{3}{2}$. \square

In fact, in the proof, when $a \leq 1$, the polynomial g always yields $g(x) \leq x$, which indicates that the fixed-point mapping $X_{k+1} = g(X_k)$ never converges to $\text{sign}(A)$. Then, when one considers only $1 < a \leq \frac{3}{2}$, g' is decreasing with respect to a . Hence, the rate of reduction in the condition number is at most $\frac{3}{2}$.

The reduction in the condition number informs only the behavior of the first iteration. Also of interest are the first few iterations. Clearly, the sequence $x_{k+1} = f(x_k)$ generated through the mapping is monotonically increasing and approaching 1. When x_0 is sufficiently small, however, the following result indicates that the first few x_k 's are also small. In particular, they depart from 0 at only a linear rate.

THEOREM 2.2. *Let f be the standard Newton-Schulz mapping (cf. (2.1)), and define a sequence $x_{k+1} = f(x_k)$ with an initial value $x_0 \in (0, 1)$. Then,*

$$\log \left(\frac{x_k}{x_0} \right) < k \log \left(\frac{3}{2} \right) < \log \frac{x_k - f^{-1}(x_k)}{\frac{1}{3}x_0 - [f^{-1}(x_k) - \frac{2}{3}x_k]} \quad (2.2)$$

whenever

$$\frac{1}{3}x_0 > \left[f^{-1}(x_k) - \frac{2}{3}x_k \right]. \quad (2.3)$$

Proof. We first note that

$$f'(x) = \frac{3}{2}(1 - x^2) > 0 \quad \text{and} \quad f''(x) = -3x < 0,$$

when $0 < x < 1$. Hence, for any k , $x_k < f'(0)x_{k-1}$. Because $f'(0) = \frac{3}{2}$, by induction we have that

$$x_k < \left(\frac{3}{2}\right)^k x_0.$$

This proves the first inequality of (2.2).

Next, we have

$$\begin{aligned} \left(\frac{3}{2}\right)^k x_0 - x_k &= \left(\frac{3}{2}\right)^{k-1} \left(\frac{3}{2}x_0 - x_1\right) + \left(\frac{3}{2}\right)^{k-2} \left(\frac{3}{2}x_1 - x_2\right) \\ &\quad + \cdots + \left(\frac{3}{2}\right)^0 \left(\frac{3}{2}x_{k-1} - x_k\right). \end{aligned}$$

Because $f'(0) = \frac{3}{2}$ and f' is decreasing, we have that $\frac{3}{2}x - f(x)$ is positive and is increasing. Then,

$$\begin{aligned} \left(\frac{3}{2}\right)^k x_0 - x_k &< \left[\left(\frac{3}{2}\right)^{k-1} + \cdots + \left(\frac{3}{2}\right)^0 \right] \left(\frac{3}{2}x_{k-1} - x_k\right) \\ &= \frac{\left(\frac{3}{2}\right)^k - 1}{\frac{3}{2} - 1} \left(\frac{3}{2}f^{-1}(x_k) - x_k\right). \end{aligned}$$

Rearranging terms, we obtain

$$\left\{ \left(\frac{3}{2} - 1\right) x_0 - \left(\frac{3}{2}f^{-1}(x_k) - x_k\right) \right\} \left(\frac{3}{2}\right)^k < \frac{3}{2} [x_k - f^{-1}(x_k)],$$

which proves the second inequality of (2.2). \square

To understand the use of Theorem 2.2, we give an example. Consider that the bound (2.2) is with respect to k . Table 2.1 gives the numeric values of (2.2) for $x_0 = 10^{-3}$, where NA means the condition (2.3) is invalid. One sees that the bound applies only when x_k is not close to 1 (otherwise (2.3) is invalid); however, whenever it is applicable, the bound for integer k is tight. Thus, we interpret the inequality on the left as

$$\log\left(\frac{x_k}{x_0}\right) \approx k \log\left(\frac{3}{2}\right) \quad (2.4)$$

for small x_k . In other words, x_k grows linearly for the first few k 's when the starting value x_0 is sufficiently small. Note that the factor $\log\left(\frac{3}{2}\right)$ is important; we will return to this factor later.

TABLE 2.1
Numerical values of (2.2) for $x_0 = 10^{-3}$.

k	1	2	3
Bound (2.2)	$0.99.. < k < 1.00..$	$1.99.. < k < 2.00..$	$2.99.. < k < 3.00..$
x_k	1.5000e-03	2.2500e-03	3.3750e-03
k	4	5	6
Bound (2.2)	$3.99.. < k < 4.00..$	$4.99.. < k < 5.00..$	$5.99.. < k < 6.00..$
x_k	5.0625e-03	7.5936e-03	1.1390e-02
k	7	8	9
Bound (2.2)	$6.99.. < k < 7.00..$	$7.99.. < k < 8.01..$	$8.99.. < k < 9.03..$
x_k	1.7085e-02	2.5624e-02	3.8428e-02
k	10	11	12
Bound (2.2)	$9.99.. < k < 10.13..$	$10.99.. < k < 11.51..$	$11.98.. < k < 14.51..$
x_k	5.7614e-02	8.6325e-02	1.2917e-01
k	13	14	15
Bound (2.2)	$12.97.. < k < \text{NA}$	$13.94.. < k < \text{NA}$	$14.87.. < k < \text{NA}$
x_k	1.9267e-01	2.8543e-01	4.1652e-01

We summarize a few facts in the following theorem, whose validity is clear based on the preceding discussions. Hence, the proof is omitted. The theorem connects the scalar iteration with the matrix iteration.

THEOREM 2.3. *For a Hermitian matrix A and the standard Newton-Schulz mapping f defined in (2.1), consider the matrix iteration $X_{k+1} = f(X_k)$, $X_0 = A$, and the scalar iteration $x_{k+1} = f(x_k)$. If the spectral radius of A is 1 and x_0 is the smallest magnitude eigenvalue of A , then we have the following.*

1. x_k is the smallest magnitude eigenvalue of X_k for all k .
2. The spectral radius of X_k is 1 for all k ; hence the condition number of X_k is $1/x_k$.
3. $\|X_k - S\|_2 = |x_k - 1|$ for all k .
4. $x_k \rightarrow 1$ monotonically and hence $\|X_k - S\|_2 \rightarrow 0$ monotonically.

3. Newton-Schulz variant. In this section, we derive a variant of Newton-Schulz so that the iteration progresses better initially. Theorem 2.1 states that the standard Newton-Schulz mapping f is optimal among all cubic and odd polynomials that map $[0, 1]$ to $[0, 1]$, if in addition the polynomial is required to be increasing. To obtain a polynomial whose derivative at the origin is even larger, we need to sacrifice the monotonicity.

Define

$$P' = \{\text{cubic and odd polynomial } h : h([0, 1]) = [0, 1]\}.$$

The polynomials in P' can be parameterized in several ways. Consider $\tilde{h}(x) = ax + bx^3$, where $a > 0$. We crop \tilde{h} in the box $[0, c] \times [0, d]$ and normalize it to obtain

$$h(x) = \frac{1}{d}\tilde{h}(cx) = \frac{ac}{d}x + \frac{bc^3}{d}x^3. \quad (3.1)$$

When d is the maximum of \tilde{h} on $[0, c]$, such polynomials h constitute the set P' .

One can separate the values of b and c into three cases and calculate that

$$d = \begin{cases} ac + bc^3, & \text{in case 1: } b \geq 0 \\ ac + bc^3, & \text{in case 2: } b < 0 \text{ and } c \leq \sqrt{-\frac{a}{3b}} \\ \frac{2a}{3} \sqrt{-\frac{a}{3b}}, & \text{in case 3: } b < 0 \text{ and } c > \sqrt{-\frac{a}{3b}}. \end{cases} \quad (3.2)$$

See Figure 3.1 for visual examples of the three cases.

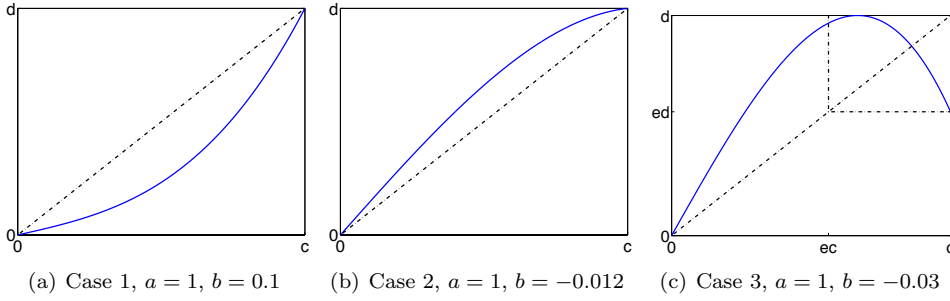


FIG. 3.1. Function \tilde{h} in different cases of (3.2). $c = 5$.

In case 1, for all $x \in [0, 1]$, $h(x) \leq x$, because the cubic polynomial $h(x) - x$ has roots $-1, 0, 1$ and on the interval $(0, 1)$ it is clearly negative. In case 2,

$$h'(0) = \frac{ac}{d} = \frac{ac}{ac + bc^3} \leq \frac{3}{2},$$

because $c \leq \sqrt{-\frac{a}{3b}}$. These two cases do not yield a better polynomial than the standard Newton-Schulz mapping f . On the other hand, in case 3,

$$h'(0) = \frac{ac}{d} = \frac{3}{2}c \left/ \sqrt{-\frac{a}{3b}} \right.,$$

which is larger than $\frac{3}{2}$. Hence, we further explore this case.

Define

$$e := \frac{ac + bc^3}{d} = \frac{ac + bc^3}{\frac{2a}{3} \sqrt{-\frac{a}{3b}}}.$$

When $x < e$, we always have $h(x) \leq x$, whereas when $x \geq e$, h maps $[e, 1]$ to $[e, 1]$ (see the boxed area on the upper right corner of Figure 3.1(c)). Thus, when one tries to increase c from $\sqrt{-\frac{a}{3b}}$ in order to maximize $h'(0)$, e monotonically decreases. As before, we let $x_0 > 0$ be the smallest magnitude eigenvalue of A , whereas the largest one is 1. We want e to be not too large, because otherwise $h(x_0)$ falls between e and 1, which consequently loses the guarantee that the condition number of the matrix is reduced. Then, by setting $h(x_0) \leq e$, we yield the constraint

$$bc^3 + ac - h(x_0) \frac{2a}{3} \sqrt{-\frac{a}{3b}} \geq 0. \quad (3.3)$$

One can easily show that the left-hand side of (3.3), as a function of c , has two real roots c_1 and c_2 such that $0 < c_1 < \sqrt{-\frac{a}{3b}} < c_2 < \sqrt{-\frac{a}{b}}$. When c is chosen from the interval $[c_1, c_2]$, the inequality (3.3) is satisfied. Then, maximizing $h'(0)$ leads to $c = c_2$, in which case (3.3) is satisfied with equality.

To yield an explicit form for c , we define $c = \alpha\sqrt{-\frac{a}{3b}}$. Then, the equality of (3.3) can be rewritten as

$$\alpha^3 - 3\alpha + 2h(x_0) = 0. \quad (3.4)$$

Meanwhile, h in (3.1) becomes

$$h(x) = \frac{3}{2}\alpha x - \frac{1}{2}\alpha^3 x^3. \quad (3.5)$$

In effect, we have replaced three parameters a , b , and c by a single parameter α . Substituting (3.5) into (3.4) gives

$$\alpha = \sqrt{\frac{3}{1 + x_0 + x_0^2}}. \quad (3.6)$$

Thus, the function h as defined in (3.5) and (3.6) is the optimal polynomial we seek from P' .

Note an important property of h : it maps the interval $[x_0, 1]$ to the interval $[h(x_0), 1]$, where $h(x_0)$ is always strictly larger than x_0 . Furthermore, $h(x_0)$ is the smallest magnitude eigenvalue of $h(A)$. In other words, h is used for updating not only the matrix but also the smallest magnitude eigenvalue of the matrix. Note also that h is not fixed; rather, it depends on the known spectral information of the current matrix throughout the iteration. This feature implies that at the beginning $x_0 \approx 0$ and we have $\alpha \approx \sqrt{3}$; hence the derivative of h at the origin is approximately $\frac{3}{2}\sqrt{3}$. On the other hand, if $x_0 \approx 1$, then $\alpha \approx 1$. In the limit, h recovers the standard Newton-Schulz mapping f .

We summarize in Algorithm 1 the calculation of $\text{sign}(A)$ by using (3.5) and (3.6) in an iterative fashion.

Algorithm 1 Newton-Schulz variant for computing $\text{sign}(A)$

- 1: Compute the smallest ($\lambda_{|\min|}$) and largest ($\lambda_{|\max|}$) absolute eigenvalue of A .
 - 2: Let $X_0 = A/\lambda_{|\max|}$ and $x_0 = \lambda_{|\min|}/\lambda_{|\max|}$.
 - 3: **for** $k = 0, 1, \dots, \text{maxiter}$ **do**
 - 4: Compute $\alpha_k = \sqrt{\frac{3}{1 + x_k + x_k^2}}$.
 - 5: Update $X_{k+1} = \frac{1}{2}\alpha_k X_k (3I - \alpha_k^2 X_k^2)$.
 - 6: Update $x_{k+1} = \frac{1}{2}\alpha_k x_k (3 - \alpha_k^2 x_k^2)$.
 - 7: If converged, exit loop.
 - 8: **end for**
 - 9: **return** X_{k+1}
-

4. Analysis. We define here the mapping for the Newton-Schulz variant proposed in the previous section:

$$h(x) = \frac{3}{2}\alpha x - \frac{1}{2}\alpha^3 x^3, \quad \text{where} \quad \alpha(x) = \sqrt{\frac{3}{1 + |x| + x^2}}. \quad (4.1)$$

Note that the mapping h defined here is not exactly the same as that in (3.5). The h in (3.5) was originally treated as a polynomial with α being a parameter. However, one sees in the above derivation that α in (3.6) actually depends on the spectral information of the matrix iterate; hence, it is no longer fixed. Thus, in (4.1), we make explicit the dependence of α on x and use (4.1) as the formal definition of h in the subsequent discussion. Figure 4.1 plots h on $[-1, 1]$, together with the standard Newton-Schulz mapping f for comparison.

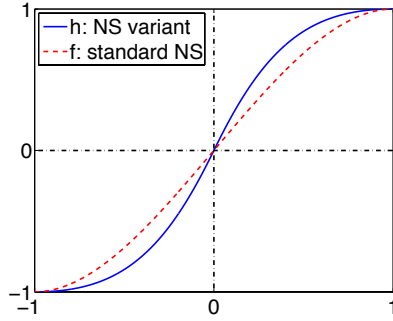


FIG. 4.1. Mapping h (Newton-Schulz variant) and f (standard Newton-Schulz).

We note that different from the case of f , we shall not treat h as a matrix function and write $X_{k+1} = h(X_k)$, because α takes only a scalar value x as input. By construction, the smallest magnitude eigenvalue maintains its “smallest” property after one iteration. Hence, we define the matrix counterpart of h as

$$\underline{h}(X) = \frac{3}{2}\underline{\alpha}X - \frac{1}{2}\underline{\alpha}^3 X^3, \quad \text{where} \quad \underline{\alpha}(X) = \sqrt{\frac{3}{1 + \lambda_{|\min|}(X) + \lambda_{|\min|}(X)^2}}. \quad (4.2)$$

Then, the iteration $X_{k+1} = \underline{h}(X_k)$ is consistent with that of Algorithm 1.

Clearly, the mapping h is monotonically increasing on $[0, 1]$, and it maps this interval to itself. Because $\alpha \geq 1$, we always have $h(x) \geq f(x)$ for $x \in (0, 1)$. Hence, for the same initial value $x_0 = \tilde{x}_0 \in (0, 1)$, the sequence $x_{k+1} = h(x_k)$ is always larger than the sequence $\tilde{x}_{k+1} = f(\tilde{x}_k)$, elementwise. Because $\tilde{x}_k \rightarrow 1$ and x_k is bounded by 1, the sequence x_k monotonically increases to the limit 1, the same as does the sequence \tilde{x}_k . Furthermore, x_k is always closer to the limit than is \tilde{x}_k . We summarize this result, together with other facts in the following theorem, whose proof is clear based on the foregoing discussion. This theorem connects the matrix iteration with the scalar iteration. One should compare this result with Theorem 2.3.

THEOREM 4.1. *For a Hermitian matrix A and the Newton-Schulz variant mappings h and \underline{h} defined in (4.1) and (4.2), respectively, consider the matrix iteration $X_{k+1} = \underline{h}(X_k)$, $X_0 = A$, and the scalar iteration $x_{k+1} = h(x_k)$. If the spectral radius of A is 1 and x_0 is the smallest magnitude eigenvalue of A , then we have the following.*

1. x_k is the smallest magnitude eigenvalue of X_k for all k .
2. $\|X_k - S\|_2 = |x_k - 1|$ for all k .
3. $x_k \rightarrow 1$ monotonically and hence $\|X_k - S\|_2 \rightarrow 0$ monotonically.
4. The spectral radius of X_k converges to 1.

Furthermore, for the iteration $\tilde{x}_{k+1} = f(\tilde{x}_k)$ where $\tilde{x}_0 = x_0$ and where f is the standard Newton-Schulz mapping, we have $\tilde{x}_k < x_k$ for all $k > 0$.

The significance of the first conclusion of Theorem 4.1 is that the convergence behavior of X_k is completely characterized by that of x_k . Then, we need to focus on only the mapping h . The following theorem states the limiting and the initial behavior of the Newton-Schulz variant.

THEOREM 4.2. *Let h be the mapping of the Newton-Schulz variant (cf. (4.1)), and define a sequence $x_{k+1} = h(x_k)$ with an initial value $x_0 \in (0, 1)$. Then*

1. x_k converges to 1 quadratically; and
2. we have

$$\log \left(\frac{x_k}{x_0} \right) < k \log \left(\frac{3}{2} \sqrt{3} \right) < \log \frac{x_k - h^{-1}(x_k)}{\left(1 - \frac{2}{3\sqrt{3}}\right) x_0 - \left[h^{-1}(x_k) - \frac{2}{3\sqrt{3}} x_k\right]} \quad (4.3)$$

whenever

$$\left(1 - \frac{2}{3\sqrt{3}}\right) x_0 > \left[h^{-1}(x_k) - \frac{2}{3\sqrt{3}} x_k\right].$$

Proof. We already know that x_k converges to 1 in Theorem 4.1. Because $h(x) - 1 = -\frac{1}{2}(\alpha x + 2)(\alpha x - 1)^2$, we have

$$\lim_{x \rightarrow 1} \frac{|h(x) - 1|}{|x - 1|^2} = \left(\lim_{x \rightarrow 1} \frac{|\alpha x + 2|}{2} \right) \left(\lim_{x \rightarrow 1} \frac{\alpha x - 1}{x - 1} \right)^2 = \frac{3}{2} \left(\lim_{x \rightarrow 1} \frac{d\alpha}{dx} + \alpha \right)^2 = \frac{3}{8}, \quad (4.4)$$

where the second equality follows from L'Hospital's rule. This shows that the convergence is quadratic.

To prove (4.3), we first note that

$$h'(x) = \frac{3}{2}(\alpha x)'(1 - \alpha^2 x^2) \quad \text{and} \quad h''(x) = \frac{3}{2}(\alpha x)''(1 - \alpha^2 x^2) - 3\alpha x[(\alpha x)']^2.$$

Because $(\alpha x)' > 0$, $(\alpha x)'' < 0$ and $1 - \alpha^2 x^2 > 0$ when $0 < x < 1$, h' is always positive whereas h'' is negative. Hence, for any k , $x_k < h'(0)x_{k-1}$. Clearly, $h'(0) = \frac{3}{2}\alpha|_{x=0}$. Thus, by induction,

$$x_k < \left(\frac{3}{2}\sqrt{3}\right)^k x_0,$$

which proves the first inequality of (4.3).

Furthermore, we have

$$\begin{aligned} \left(\frac{3}{2}\sqrt{3}\right)^k x_0 - x_k &= \left(\frac{3}{2}\sqrt{3}\right)^{k-1} \left(\frac{3}{2}\sqrt{3}x_0 - x_1\right) \\ &\quad + \left(\frac{3}{2}\sqrt{3}\right)^{k-2} \left(\frac{3}{2}\sqrt{3}x_1 - x_2\right) + \cdots + \left(\frac{3}{2}\sqrt{3}\right)^0 \left(\frac{3}{2}\sqrt{3}x_{k-1} - x_k\right). \end{aligned}$$

Because $h'(0) = \frac{3}{2}\sqrt{3}$ and h' is decreasing, we have that $\frac{3}{2}\sqrt{3}x - h(x)$ is positive and is increasing. Then,

$$\begin{aligned} \left(\frac{3}{2}\sqrt{3}\right)^k x_0 - x_k &< \left[\left(\frac{3}{2}\sqrt{3}\right)^{k-1} + \cdots + \left(\frac{3}{2}\sqrt{3}\right)^0 \right] \left(\frac{3}{2}\sqrt{3}x_{k-1} - x_k\right) \\ &= \frac{\left(\frac{3}{2}\sqrt{3}\right)^k - 1}{\frac{3}{2}\sqrt{3} - 1} \left(\frac{3}{2}\sqrt{3}h^{-1}(x_k) - x_k\right). \end{aligned}$$

Rearranging terms, we obtain

$$\left\{ \left(\frac{3}{2}\sqrt{3} - 1\right) x_0 - \left(\frac{3}{2}\sqrt{3}h^{-1}(x_k) - x_k\right) \right\} \left(\frac{3}{2}\sqrt{3}\right)^k < \frac{3}{2}\sqrt{3} [x_k - h^{-1}(x_k)],$$

which proves the second inequality of (4.3). \square

Two relevant points are noted. First, even though both mappings f and h converge to 1 quadratically as $x \rightarrow 1$, the convergence of h is “faster.” This claim is in the sense that the limit (4.4) can be used to establish the quadratic convergence of h , but a similar limit cannot be used for that of f , because for f we have

$$\lim_{x \rightarrow 1} \frac{|f(x) - 1|}{|x - 1|^2} = \frac{3}{2},$$

which is larger than 1. The quadratic convergence of f can be shown by using a technique similar to (and in fact simpler than) that of the proof of Theorem 1.1.

Second, similar to the interpretation of Theorem 2.2, we see that the bound (4.3) is tight. For example, when $x_0 = 10^{-3}$, the numeric values of the bound are given in Table 4.1. Hence, we write

$$\log\left(\frac{x_k}{x_0}\right) \approx k \log\left(\frac{3}{2}\sqrt{3}\right). \quad (4.5)$$

Compare (4.5) with (2.4), which we repeat here by adding a tilde to denote the sequence generated through the f mapping (as we previously did):

$$\log\left(\frac{\tilde{x}_k}{x_0}\right) \approx k \log\left(\frac{3}{2}\right).$$

Because the ratio between $\log\left(\frac{3}{2}\sqrt{3}\right)$ and $\log\left(\frac{3}{2}\right)$ is 2.35..., we can loosely conclude that

$$\tilde{x}_{2k} < x_k < \tilde{x}_{3k}.$$

This means that initially (when k is small), the iteration of Newton-Schulz variant increases the iterate x_k at least twice as fast as does the standard Newton-Schulz iteration.

We empirically compare the convergence of $x_{k+1} = h(x_k)$ and $\tilde{x}_{k+1} = f(\tilde{x}_k)$ for the same initial value $x_0 = \tilde{x}_0$. Figure 4.2 shows a few convergence curves for different x_0 . All iterations are run to machine precision. In general, the number of iterations for h is only half of that for f . The quadratic decay of errors in both iterations is clearly seen when x_{k+1} and \tilde{x}_{k+1} are in the proximity of the limit. The advantage of the h mapping lies in the faster decrease of error in the initial stage.

TABLE 4.1
Numerical values of (4.3) for $x_0 = 10^{-3}$.

k	1	2	3
Bound (4.3)	$0.99.. < k < 1.00..$	$1.99.. < k < 2.00..$	$2.99.. < k < 3.03..$
x_k	2.5968e-03	6.7378e-03	1.7445e-02
k	4	5	6
Bound (4.3)	$3.98.. < k < 4.28..$	$4.95.. < k < \text{NA}$	$5.88.. < k < \text{NA}$
x_k	4.4914e-02	1.1383e-01	2.7539e-01

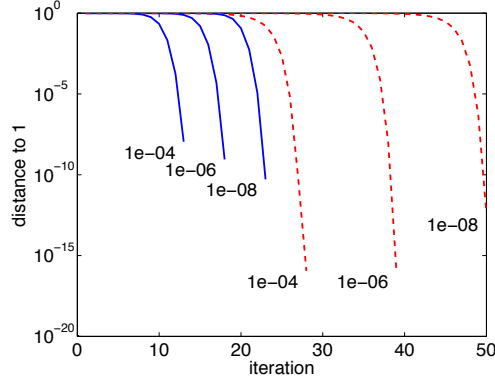


FIG. 4.2. Convergence of h mapping (blue, solid) and f mapping (red, dashed). Numbers for each curve indicate the initial value x_0 . For each case, one iteration further, the distance to 1 is either 0 or less than machine precision.

On closing this section, we remark that in the convergence guarantee of Algorithm 1, x_0 need not be the smallest magnitude eigenvalue of X_0 . The following theorem states that Algorithm 1 always converges no matter what value x_0 takes. What is more amazing is that the quadratic rate of convergence is also maintained. Hence, the price paid for using an arbitrary x_0 is only some more iterations.

THEOREM 4.3. *In Algorithm 1, X_k converges to S quadratically for any $x_0 \in (0, 1)$.*

Proof. We repeat the iteration of Algorithm 1 in the following:

$$X_{k+1} = \frac{1}{2}\alpha_k X_k (3I - \alpha_k^2 X_k^2). \quad (4.6)$$

Because the sequence x_k is computed through the fixed-point mapping $x_{k+1} = h(x_k)$, for any $x_0 \in (0, 1)$, x_k converges to 1 quadratically. Hence, α_k also converges to 1 quadratically. We use an auxiliary sequence

$$Y_{k+1} = \frac{1}{2}Y_k (3I - Y_k^2), \quad Y_0 = X_0 \quad (4.7)$$

to gauge the convergence behavior of X_k . Clearly, the sequence Y_k results from the standard Newton-Schulz iteration, and it converges to S quadratically.

Let $V_k = X_k - Y_k$, $W_k = Y_k - S$, and $\epsilon_k = |\alpha_k - 1|$. We have

$$\begin{aligned} \|\alpha_k X_k - Y_k\|_2 &\leq \|\alpha_k X_k - \alpha_k Y_k\|_2 + \|\alpha_k Y_k - Y_k\|_2 \\ &= \alpha_k \|V_k\|_2 + \epsilon_k \|Y_k\|_2 \leq (1 + \epsilon_k) \|V_k\|_2 + \epsilon_k (1 + \|W_k\|_2). \end{aligned} \quad (4.8)$$

We further let $Z_k = \alpha_k X_k - Y_k$. By noting that X_k and Y_k commute (because both are polynomials of X_0), we subtract (4.7) from (4.6) and obtain

$$\begin{aligned} \|V_{k+1}\|_2 &= \frac{1}{2} \|Z_k(3I - Z_k^2 - 3\alpha_k X_k Y_k)\|_2 \\ &\leq \frac{1}{2} \|Z_k\|_2 \left(3\|I - Y_k^2\|_2 + \|Z_k\|_2^2 + 3\|Z_k\|_2 \|Y_k\|_2 \right) \\ &\leq \frac{1}{2} \|Z_k\|_2 \left(\|Z_k\|_2^2 + 3(1 + \|W_k\|_2)\|Z_k\|_2 + 3(2\|W_k\|_2 + \|W_k\|_2^2) \right). \end{aligned}$$

With the help of (4.8) we further expand the inequality of $\|V_{k+1}\|_2$ and obtain

$$\begin{aligned} \|V_{k+1}\|_2 &\leq \left[3(\epsilon_k + \|W_k\|_2) + O(\epsilon_k^2 + \epsilon_k \|W_k\|_2 + \|W_k\|_2^2) \right] \cdot \|V_k\|_2 \\ &\quad + O(\epsilon_k^2 + \epsilon_k \|W_k\|_2 + \|W_k\|_2^2 + \|V_k\|_2^2). \end{aligned}$$

Note that both ϵ_k and $\|W_k\|_2$ converge to 0 quadratically. Because $\|V_0\|_2 = 0$, by induction we have that $\|V_k\|_2 = O(\epsilon_k + \|W_k\|_2)$. This means that the difference between X_k and Y_k converges to 0 quadratically. Thus, X_k converges to S quadratically. \square

5. Comparison with Newton's Method. The mapping for Newton's method is

$$p(x) = \frac{1}{2} \left(x + \frac{1}{x} \right), \quad (5.1)$$

and hence Newton's iteration reads $X_{k+1} = p(X_k)$, $X_0 = A$. One advantage of Newton's method is that it is globally convergent; thus, eigenvalue estimation is unnecessary. Nevertheless, we note that several techniques have been proposed to scale Newton's iteration in order to improve convergence (see [9, Section 5.5]). In some of these techniques, spectral information is computed and exploited.

In this section, we compare the convergence of Newton's method with that of Newton-Schulz and the variant. The comparison can be made convenient by assuming that A is Hermitian, in which case the convergence of Newton's method is completely characterized by either the largest or the smallest magnitude eigenvalue of A undergoing the fixed-point mapping p . To be precise, we define the scalar function

$$\mathcal{I}(x) = \begin{cases} x, & |x| \geq 1 \\ x^{-1}, & |x| < 1, x \neq 0. \end{cases}$$

The matrix function $\mathcal{I}(A)$ is well defined for a nonsingular A . Clearly, the Newton sequence X_1, X_2, \dots generated from $X_0 = A$ is exactly the same as that generated from $X_0 = \mathcal{I}(A)$. The reason we consider $\mathcal{I}(A)$ is that all its eigenvalues have a magnitude larger than or equal to 1. Note that $p(x)$ is increasing when $x \geq 1$ and $p(x) \geq 1$ in such a case. Then, the largest magnitude eigenvalue of X_k monotonically decreases to the limit 1. Therefore, if we let $x_0 = \lambda_{|\max|}(\mathcal{I}(A))$ and $x_{k+1} = p(x_k)$, then

$$\|X_k - S\|_2 = |x_k - 1|$$

for all k . In other words, the convergence of the matrix sequence X_k is the same as that of the scalar sequence x_k .

In Figure 5.1, we repeat Figure 4.2 but overlay it with the convergence curve for the p mapping for different initial values x_0 . Note that for Newton-Schulz (f mapping) and the variant (h mapping), x_0 is less than 1, whereas for Newton (p mapping), a starting value x_0 produces the same sequence as does x_0^{-1} . Hence, we assume that all x_0 are smaller than 1. Clearly, the smaller the x_0 , the more iterations are required for convergence. One sees that Newton converges faster than Newton-Schulz but slower than the Newton-Schulz variant.

One must be cautious when using the convergence of the scalar sequence to interpret the convergence of the matrix sequence, even though the two are closely related. For Newton-Schulz and the variant, the starting value x_0 corresponds to the smallest magnitude eigenvalue of the scaled matrix, that is, $\lambda_{|\min|}(A)/\lambda_{|\max|}(A)$, whereas for Newton, the starting value corresponds to the reciprocal of the largest magnitude eigenvalue of $\mathcal{I}(A)$, that is, $\lambda_{|\max|}(\mathcal{I}(A))^{-1}$. As a side note, we mention that one simple way to improve the convergence of Newton is to prescale the matrix A such that the product of its largest magnitude eigenvalue and the smallest magnitude eigenvalue is 1. This prescaling maximally reduces $\lambda_{|\max|}(\mathcal{I}(A))$.

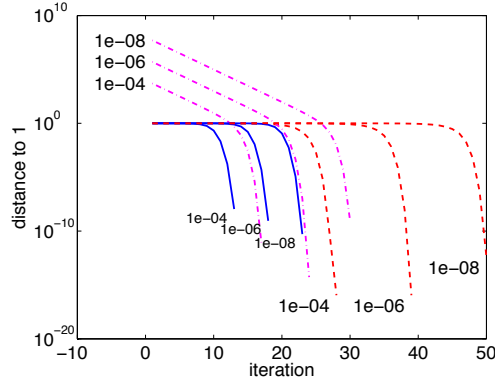


FIG. 5.1. Figure 4.2 overlaid with the convergence of p mapping (dashed-dotted).

6. Practical implementation. A few implementation issues for Algorithm 1 are addressed in this section.

6.1. Stopping criterion. We need a convergence test for line 7 of Algorithm 1. We consider two methods. In the first method, we write $X_k - S = (X_k S - I)S$ and hence

$$\|X_k - S\| = \|(X_k S - I)S\| \leq \|X_k S - I\| \|S\|.$$

Note that the inequality is an equality in the 2-norm case because $X_k S - I$ and S can be simultaneously and unitarily diagonalized and the eigenvalues of S are either 1 or -1 . When $X_k \approx S$, we obtain the relative error

$$\frac{\|X_k - S\|}{\|S\|} \lesssim \|X_k^2 - I\|.$$

Hence, it is straightforward to use $\|X_k^2 - I\| \leq \epsilon$ as the convergence criterion, where ϵ is the relative tolerance. Any submultiplicative norm can be used, but for computational

convenience we can use the Frobenius norm. The computed squared factor X_k^2 is stored and used in the next iteration for updating X_k .

In the second method, the absolute (and equivalently, the relative) 2-norm error $\|X_k - S\|_2 = |x_k - 1|$, according to Theorem 4.1. Then, the convergence criterion can be simply set as $|x_k - 1| \leq \epsilon$. Note that if the starting value x_0 is arbitrary (as mentioned at the end of Section 4), this criterion is invalid.

6.2. Shifting. A useful observation is that shifts of A do not change $\text{sign}(A)$ as long as these shifts do not cause eigenvalues of A to cross the origin. Hence, we can perform a shift to precondition A (that is, to start with a larger x_0) and to improve convergence. Specifically, we first compute four eigenvalues of interest:

$$\lambda_{-\min}(A) \leq \lambda_{-\max}(A) < 0 < \lambda_{+\min}(A) \leq \lambda_{+\max}(A),$$

which are the smallest negative, the largest negative, the smallest positive, and the largest positive eigenvalue of A , respectively, and compute a shift

$$\tau = \frac{\lambda_{-\max}(A) + \lambda_{+\min}(A)}{2}.$$

Then, we input $A' = A - \tau$ and run Algorithm 1 to obtain $S = \text{sign}(A')$. The two eigenvalues in Algorithm 1 now become

$$\begin{aligned} \lambda'_{|\max|} &= \max\{|\lambda_{+\max}(A) - \tau|, |\lambda_{-\min}(A) - \tau|\} \\ \lambda'_{|\min|} &= \min\{|\lambda_{-\max}(A) - \tau|, |\lambda_{+\min}(A) - \tau|\}. \end{aligned}$$

Figure 4.2 gives an intuitive estimate of the amount of reduction in iterations because of the reduction in condition number.

6.3. Eigenvalue estimation. The requirement of eigenvalue computation in Algorithm 1 forms several levels of difficulty. Our discussion here is oriented toward general eigenvalue techniques and software support. Whereas small-scale eigenvalue problems have a standard solution through orthogonal transformations to the condensed form, large-scale eigenvalue problems are challenging, even if only a few extreme and/or interior eigenvalues are sought. We make no effort in designing new eigenvalue techniques; rather, the issue we address here is how practical Algorithm 1 is in a large-scale setting with off-the-shelf software packages. For a comprehensive treatment of the theory and state-of-the-art eigenvalue methods, we refer the readers to Saad's book [20]. Practitioners might develop specialized eigenvalue solvers for their applications.

At the easiest level, no eigenvalue computation is carried out. The largest magnitude eigenvalue can be estimated by using the Gershgorin circle theorem. The theorem ensures that the spectral radius of the scaled matrix is no greater than 1. A drawback of this method is that in some cases the bound provided by the theorem is too pessimistic. On the other hand, the starting value x_0 can be arbitrary, as noted earlier; what is sacrificed is the optimal convergence. If a good estimate of the condition number of A is known a priori, the extra number of iterations may not be too large. Note that if x_0 is not an accurate estimate, one must use the first stopping criterion proposed in Section 6.1. Naturally, shifting is impossible.

At the next level, Gershgorin is replaced by a computation of the largest magnitude eigenvalue. This eigenvalue can be computed by using the Lanczos algorithm. Accelerated by implicit shifts and restarts [12, 22], the Lanczos algorithm converges

rapidly when the targeted eigenvalue is not clustered with others. The algorithm has been implemented in ARPACK [13]. The dominant cost of the calculation is forming matrix-vector products, for which extensive research has been devoted to designing high-performance software as well as linear- or near linear-time algorithms for different types of matrices, such as sparse, Toeplitz, or kernel matrices. Because the desired eigenvalue converges from inside the spectrum interval, if this eigenvalue cannot be computed to high accuracy, a correction term must be added to ensure that the spectral radius of the scaled matrix is no greater than 1.

The most difficult level comprises the calculation of all four eigenvalues mentioned in Section 6.2. In particular, shifting is useful and produces the correct result only when the two innermost eigenvalues are computed accurately. Computing these eigenvalues, however, is a well-known challenge in applications. The Lanczos algorithm discussed in the preceding paragraph can be reused, by applying A^{-1} as the operator instead of A . Then, the dominant cost becomes solving linear systems with A . For sparse matrices, a direct method is the most robust; software includes CHOLMOD [3] and SuperLU [15]. For dense matrices arising from kernels, direct solves with high-accuracy off-diagonal compression techniques are gaining popularity [25]. Iterative solvers are in general not as robust as direct solvers but sometimes are highly efficient with a good preconditioner. To avoid being too restricted by the standard form of Lanczos, we also mention other popular methods for computing interior eigenvalues: shift-and-invert [5], Davidson's method [2] and its improvements [18, 7], and polynomial filtering [6], the details of which are not further discussed.

7. Numerical experiments. In this section we demonstrate numerical calculations with two sets of matrices: one from a toy problem and the other from the application of density functional theory in quantum mechanical modeling. Several aspects of the proposed algorithm are examined, including the accurate/inaccurate estimates of eigenvalues, shifting, and different computing environments including a single desktop with multithreaded Matlab and a computing cluster with MPI. The goal of the experiments is to demonstrate the superiority of the proposed algorithm compared with standard Newton-Schulz and the appealing scalability of an algorithm that is multiplication-rich.

7.1. Toy problem. We consider a matrix built using the standard Laplacian L :

$$A = \begin{bmatrix} L - c\lambda_{|\min|}(L) & \\ & -2L + 2c\lambda_{|\min|}(L) \end{bmatrix}, \quad (7.1)$$

where $c < 1$ is a tunable parameter. Here, L is the 2D Laplacian defined on an $m_1 \times m_2$ grid, with known eigenvalues

$$4 \left[\sin^2 \left(\frac{i\pi}{2(m_1 + 1)} \right) + \sin^2 \left(\frac{j\pi}{2(m_2 + 1)} \right) \right], \quad i = 1, \dots, m_1, \quad j = 1, \dots, m_2.$$

Clearly,

$$\lambda_{|\max|}(A) = 2[\lambda_{|\max|}(L) - c\lambda_{|\min|}(L)] \quad \text{and} \quad \lambda_{|\min|}(A) = (1 - c)\lambda_{|\min|}(L).$$

When $c \rightarrow 1$, A is increasingly ill-conditioned.

We let $m_1 = 20$ and $m_2 = 30$, and we test with several choices of c . Table 7.1 lists the number of iterations with different estimates of the eigenvalues $\lambda_{|\max|}(A)$ and $\lambda_{|\min|}(A)$ in Algorithm 1; $\lambda_{|\max|}(A)$ is also used for scaling the matrix in Newton-Schulz. For the estimated eigenvalues, we let $\lambda_{|\max|}(A)$ be twice the exact value.

Usually, this eigenvalue is easy to estimate and setting it to be twice as large is sufficiently conservative. On the other hand, for $\lambda_{|\min|}(A)$, we perturb the exact value by a factor of 10 or 100 to simulate highly inaccurate estimates. In the table, the first row always uses the exact eigenvalues. The column “NS” stands for Newton-Schulz and “NSv” stands for the variant. All computations were run to $\|X_{k+1}^2 - I\|_F \leq 1\text{e-}14$.

TABLE 7.1

Number of iterations for matrix A in (7.1). “NS” stands for Newton-Schulz and “NSv” stands for the proposed variant (Algorithm 1). “Est. $\lambda_{|\max|}$ ” and “Est. $\lambda_{|\min|}$ ” are perturbed eigenvalues of A , except in the first row, which uses the exact eigenvalues.

$c = 0$, cond = 488.802					$c = 1 - 10^{-2}$, cond = 48682.2				
	Est. $\lambda_{ \max }$	Est. $\lambda_{ \min }$	NS	NSv		Est. $\lambda_{ \max }$	Est. $\lambda_{ \min }$	NS	NSv
1:	15.9348	3.26e-02	21	11	15.8703	3.26e-04	32	16	
2:	15.9348	1.00e-02	21	13	15.8703	1.00e-04	32	17	
3:	15.9348	1.00e-01	21	14	15.8703	1.00e-02	32	22	
4:	15.9348	1.00e-03	21	15	15.8703	1.00e-06	32	22	
5:	31.8696	3.26e-02	22	12	31.7405	3.26e-04	34	17	
6:	31.8696	1.00e-02	22	13	31.7405	1.00e-04	34	18	
7:	31.8696	1.00e-01	22	14	31.7405	1.00e-02	34	22	
8:	31.8696	1.00e-03	22	16	31.7405	1.00e-06	34	23	

$c = 1 - 10^{-4}$, cond = 4.86802e+06					$c = 1 - 10^{-6}$, cond = 4.86802e+08				
	Est. $\lambda_{ \max }$	Est. $\lambda_{ \min }$	NS	NSv		Est. $\lambda_{ \max }$	Est. $\lambda_{ \min }$	NS	NSv
1:	15.8696	3.26e-06	43	21	15.8696	3.26e-08	55	26	
2:	15.8696	1.00e-06	43	22	15.8696	1.00e-08	55	27	
3:	15.8696	1.00e-04	43	27	15.8696	1.00e-06	55	31	
4:	15.8696	1.00e-08	43	27	15.8696	1.00e-10	55	32	
5:	31.7392	3.26e-06	45	22	31.7392	3.26e-08	56	26	
6:	31.7392	1.00e-06	45	23	31.7392	1.00e-08	56	28	
7:	31.7392	1.00e-04	45	27	31.7392	1.00e-06	56	32	
8:	31.7392	1.00e-08	45	28	31.7392	1.00e-10	56	33	

A few observations follow. As expected, the worse the conditioning, the more the number of iterations. Across all test cases, when the eigenvalue estimates are accurate, one sees that the proposed variant requires only half the iterations that Newton-Schulz needs. For the variant, when both estimated eigenvalues are not far from the accurate ones, the number of iterations increases by only one or two. When the estimates are highly inaccurate, however, the number of iterations significantly increases. Nevertheless, the variant still requires much fewer iterations than does standard Newton-Schulz. The latter is relatively stable in the number of iterations because $\lambda_{|\min|}(A)$ is irrelevant to the algorithm.

7.2. PARSEC matrices. We test our algorithm on the PARSEC collection of matrices arising from density functional theory in quantum mechanics. The matrices can be downloaded from the University of Florida Sparse Matrix Collection.¹ This collection contains matrices of size ranging from several hundreds to a few hundred thousands. We use the Blues computing cluster² at Argonne National Laboratory to demonstrate the calculations for the larger matrices. The machine comprises 310 compute nodes, each of which has 16 Intel Sandy Bridge cores and 64 GB of memory. The compute nodes are connected by QLogic QDR InfiniBand with a fat-tree topology.

¹<http://www.cise.ufl.edu/research/sparse/matrices/>

²<http://www.lcrc.anl.gov/about/blues>

We prepared two programs, one in Matlab and the other in C with MPI. The Matlab program runs on one compute node with a maximum of 16 threads by default. It uses the backslash command for solving linear systems and uses the `eigs` command for computing eigenvalues. The C program runs on a large number of compute nodes. It uses SuperLU for solving linear systems and uses PARPACK (parallel ARPACK) for computing eigenvalues. The matrix-matrix multiplication is implemented naively (without tuning or optimization) by using the Cannon's algorithm, where the local matrix blocks are multiplied by using the `dgemm` kernel and the matrix blocks are communicated with `MPI_Sendrecv`.

We run the standard Newton-Schulz iteration and the proposed variant on the PARSEC matrices. The results are shown on the top part of Table 7.2. The tolerance is set as $n\mathbf{u}/2$, the best attainable relative error [9, Section 5.1], where \mathbf{u} is the machine precision. (For the two largest problems, iterations were carried out only at a later stage of the experiment.) As expected, the proposed variant in general takes half the number of iterations as required by standard Newton-Schulz. The computed inertias are highly skewed: only a tiny portion of the eigenvalues is to the left of the origin.

Next, we center the matrices at $1/3$ of the original spectrum (that is, the new center is located at $\frac{1}{3}\lambda_{+\max} + \frac{2}{3}\lambda_{-\min}$), and reperform the calculations. The cutoff $1/3$ is arbitrary; our purpose is to demonstrate calculations with an arbitrary change of the origin. The results are shown in the middle part of Table 7.2. Interestingly, in general the new inertias have a positive to negative ratio of approximately 2:1, which is aligned with the magnitude ratio between the new $\lambda_{+\max}$ and $\lambda_{-\min}$. The only exception is the matrix **Ga3As3H12**, which we discuss later. Again, the number of iterations for the Newton-Schulz variant is in general half that of standard Newton-Schulz.

We further shift the centered matrices for preconditioning while preserving the inertia, by using the formulas in Section 6.2. The results are shown in the bottom part of Table 7.2. By inspecting the changes of the eigenvalues $\lambda_{-\max}$ and $\lambda_{+\min}$, one sees that for all matrices the condition number is reduced. Hence, the iteration counts are reduced accordingly. Whereas for most of the cases the reduction is small, one particular case that yields a significant improvement is the matrix **benzene**. The results demonstrate that shifting always helps; the gain can sometimes be substantial.

One peculiar case is the matrix **Ga3As3H12**. After centering, the matrix becomes very well conditioned, and only one eigenvalue lies on the right of the origin. Then, after a further shifting, a small number of iterations are needed for both Newton-Schulz and the variant to converge.

We show in Table 7.3 the timings of the calculations for the matrices after centering and optimal shifting. Results on the top part of the table are obtained by running the Matlab program, those in the bottom part by running the C program. As expected, computing the interior eigenvalues $\lambda_{-\max}$ and $\lambda_{+\min}$ is more costly than computing the exterior eigenvalues $\lambda_{-\min}$ and $\lambda_{+\max}$. Nevertheless, compared with the iterations, the time for computing the eigenvalues is only a small portion, and this portion is almost negligible for large matrices. Note, however, that although the PARSEC matrices are sparse, we implement the iterations with dense matrix-matrix multiplications.

8. Application: electronic-structure calculation. At every iteration of the Hartree-Fock algorithm, also known as the self-consistent field (SCF) iteration, a spectral projector called the density matrix is computed from the Fock matrix, which is an approximation to the Hamiltonian [23]. The density matrix may be computed

TABLE 7.2

Computation results of the matrices in the PARSEC collection. Inertia a/b means a positive eigenvalues and b negative eigenvalues. The last two columns are the number of iterations.

Original matrix								
Matrix	n	$\lambda_{- \min}$	$\lambda_{- \max}$	$\lambda_{+ \min}$	$\lambda_{+ \max}$	Inertia	NS	NSv
Si2	769	-3.8e-1	-3.8e-1	2.4e-1	4.1e+1	768/1	18	10
SiH4	5,041	-1.0e+0	-6.3e-1	3.5e-2	3.7e+1	5,037/4	22	12
benzene	8,219	-7.3e-1	-1.7e-1	4.0e-2	5.8e+1	8,217/2	23	12
Si10H16	17,077	-1.2e+0	-1.5e-2	6.6e-4	3.7e+1	17,036/41	32	16
SiO	33,401	-1.7e+0	-2.4e-2	7.4e-2	8.4e+1	33,393/8	26	13
Ga3As3H12	61,349	-1.2e+0	-3.0e-2	1.0e-4	1.3e+3	not computed		
Si34H36	97,569	-1.2e+0	-6.4e-3	9.0e-4	4.3e+1	not computed		
Matrix centered to 1/3 of the spectrum								
Matrix	n	$\lambda_{- \min}$	$\lambda_{- \max}$	$\lambda_{+ \min}$	$\lambda_{+ \max}$	Inertia	NS	NSv
Si2	769	-1.4e+1	-1.1e-2	2.9e-2	2.8e+1	516/253	25	13
SiH4	5,041	-1.3e+1	-1.7e-3	2.0e-2	2.5e+1	3,467/1,574	29	15
benzene	8,219	-2.0e+1	-5.0e-3	7.2e-6	3.9e+1	5,459/2,760	43	21
Si10H16	17,077	-1.3e+1	-1.3e-3	6.9e-4	2.5e+1	11,575/5,502	31	16
SiO	33,401	-2.9e+1	-1.7e-3	3.1e-3	5.7e+1	22,620/10,781	31	16
Ga3As3H12	61,349	-4.3e+2	-3.3e+2	8.7e+2	8.7e+2	not computed		
Si34H36	97,569	-1.5e+1	-2.0e-5	8.9e-4	2.9e+1	not computed		
Matrix centered to 1/3 of the spectrum plus optimal shift								
Matrix	n	$\lambda_{- \min}$	$\lambda_{- \max}$	$\lambda_{+ \min}$	$\lambda_{+ \max}$	Inertia	NS	NSv
Si2	769	-1.4e+1	-2.0e-2	2.0e-2	2.8e+1	516/253	23	12
SiH4	5,041	-1.3e+1	-1.1e-2	1.1e-2	2.5e+1	3,467/1,574	24	13
benzene	8,219	-2.0e+1	-2.5e-3	2.5e-3	3.9e+1	5,459/2,760	29	15
Si10H16	17,077	-1.3e+1	-9.9e-4	9.9e-4	2.5e+1	11,575/5,502	30	15
SiO	33,401	-2.9e+1	-2.4e-3	2.4e-3	5.7e+1	22,620/10,781	30	15
Ga3As3H12	61,349	-7.0e+2	-6.0e+2	6.0e+2	6.0e+2	61,348/1	5	4
Si34H36	97,569	-1.5e+1	-4.6e-4	4.6e-4	2.9e+1	65,621/31,948	32	16

TABLE 7.3

Timing results of the computation of the matrices in Table 7.2. Only the timings for the centered matrices after optimal shifts are shown. All times are in seconds. The top part was computed on shared memory using a maximum of 16 threads; the bottom part was computed on distributed memory using the specified number of MPI processes.

Matrix	n	Parallelism	$\lambda_{- \min}$	$\lambda_{- \max}$	$\lambda_{+ \min}$	$\lambda_{+ \max}$	NS	NSv
Si2	769	16 threads	4e-1	5e-1	8e-1	5e-2	1e+0	6e-1
SiH4	5,041	16 threads	1e-1	3e+1	6e+1	9e-2	2e+2	1e+2
benzene	8,219	16 threads	2e-1	9e+1	6e+1	2e-1	9e+2	5e+2
Si10H16	17,077	16 threads	6e-1	9e+2	7e+2	6e-1	7e+3	4e+3
SiO	33,401	1,024 procs	1e+0	9e+0	7e+0	8e-1	4e+3	2e+3
Ga3As3H12	61,349	1,024 procs	4e+0	1e+2	2e+1	3e-1	3e+3	2e+3
Si34H36	97,569	2,304 procs	7e+0	4e+1	4e+1	6e+0	5e+4	2e+4

in many ways, the most obvious being through an eigenvalue decomposition of the Fock matrix. In this section we demonstrate the use of our Newton-Schulz variant for computing the density matrix.

Construction of the Fock matrix is the computationally intensive step in SCF iterations, but it is highly parallel. Computation of the density matrix, however,

TABLE 8.1

Hydrocarbon test problems. The dimension of the core-Hamiltonian matrix is the basis size, and n_{occ} is the number of occupied orbitals. The value $\lambda_{|\min|}$ is the smallest magnitude eigenvalue of A after it has been scaled to have unit spectral radius.

	Basis Size	n_{occ}	$\lambda_{ \min }$
Graphene $C_{384}H_{48}$	5,616	1,176	4.1e-05
Alkane $C_{418}H_{838}$	10,042	1,673	2.4e-05

can be the bottleneck limiting parallel scalability: although the density matrix is not extremely large, it must be efficiently computed by using a large number of processors (which were used earlier to construct the Fock matrix), much like the solution of small coarse-grid problems in parallel multigrid. Algorithms of Newton-Schulz type (called McWeeny purification [17] in the quantum chemistry literature) that are rich in matrix-matrix multiplications are thus attractive in the high parallelism setting.

We generated the core-Hamiltonian matrix for two hydrocarbon molecules: a graphene-like molecule, $C_{384}H_{48}$, and a linear alkane, $C_{418}H_{838}$. The core-Hamiltonian describes the kinetic energy and nuclear attraction of electrons, but not electron-electron interactions, and is often used as the initial guess for the Fock matrix in SCF iterations. Since the Fock matrix itself depends on the density matrix, the core-Hamiltonian initial guess corresponds to a zero initial guess for the density matrix. Here, we compute the density matrix corresponding to these core-Hamiltonians. The elements of the core-Hamiltonian matrices were formed by using the Dunning cc-pVDZ basis set [4] using a standard quantum chemistry code [16]. Table 8.1 shows the resulting matrix dimension (equal to the number of basis functions) for the two test problems.

The density matrix is the spectral projector associated with the n_{occ} lowest eigenvalues and their corresponding eigenvectors, where n_{occ} is the number of occupied orbitals, or half the number of electrons assuming closed-shell orbitals. To compute the density matrix via the sign function, we first compute the sign of

$$A = \mu I - H,$$

where H is the core-Hamiltonian in our case and μ , known as the Fermi level, separates the occupied eigenvalues from the unoccupied eigenvalues. The Fermi level is often known, especially for problems with a large energy band gap. For our tests, we chose the Fermi level to lie exactly between the n_{occ} and $n_{occ} + 1$ -st eigenvalue of A , sorted in increasing order. This is optimal for the Newton-Schulz method and the variant. In practice, such an exact shift is not known. On the other hand, using the core-Hamiltonian is a kind of worse case, since there is no gap of eigenvalues around μ , as would be the case for more converged Fock matrices in later SCF iterations. We note that once $\text{sign}(A)$ is computed, the density matrix is given by $(\text{sign}(A) + I)/2$. The density matrix can also be computed directly from A by using the McWeeny mapping $g(x) = 3x^2 - 2x^3$, which can also be modified to accelerate convergence as we have in this paper modified the Newton-Schulz mapping.

Figure 8.1 plots the residual norm $\|X_k^2 - I\|_F$ for Newton-Schulz and the variant for the two test problems. As we have seen before, the variant converges in approximately half the number of iterations of the original method. For the original method, convergence in the Frobenius norm is monotone, as every eigenvalue is improved (pushed toward the correct direction) at each iteration. Although it is faster, the situation is different for the variant method. Here, we observe a “dip” in the

first five iterations: an initial decrease, faster than the decrease of standard Newton-Schulz, followed by an increase. This observation does not contradict the monotone convergence seen in Figure 4.2, which plots a residual in the 2-norm. In the variant method, only the eigenvalue closest to zero is guaranteed to improve monotonically; the nonmonotone convergence of other eigenvalues makes the Frobenius norm of the residual converge nonmonotonically.

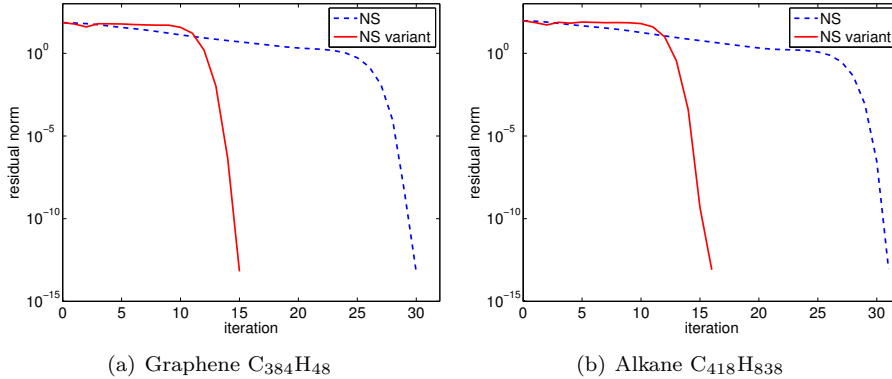


FIG. 8.1. Convergence of Newton-Schulz (NS) and the Newton-Schulz variant (NS variant) for the graphene and alkane core-Hamiltonians.

9. Conclusion. The multiplication-rich property of the Newton-Schulz method makes it preferable over other methods for computing the sign function of a Hermitian matrix in the setting of high-performance computing. In this paper, we have presented an improved variant that accelerates the initially slow convergence of Newton-Schulz. The variant improves the 2-norm residuals during the iteration over those of Newton-Schulz, and it generally requires only half the number of iterations to converge. The main idea behind the improvement is the redesign of the Newton-Schulz mapping so that eigenvalues with small magnitudes converge faster to their limits. Whereas the Newton-Schulz mapping is a polynomial, the mapping for the variant is not. On the other hand, both mappings maintain the property that the 2-norm difference between the iterate X_k and the limit S is equal to the absolute difference between a scalar iterate x_k and the limit 1, where x_k is guaranteed to approach the limit monotonically and quadratically.

The proposed variant requires estimation of the largest magnitude eigenvalue, as does the standard Newton-Schulz method. On the other hand, most of the convergence theory for the proposed variant is based on an accurate calculation of the smallest magnitude eigenvalue as well. Nevertheless, we have proved a result stating that the quadratic convergence is maintained even if the “smallest magnitude eigenvalue” required by the algorithm is blindly set, and we have demonstrated numerical results that validate the claim with the arbitrarily set value differing from the true eigenvalue by a factor of 100. The price paid is a larger number of iterations. Finally, the best performance of the proposed variant is achieved by accurately calculating all the following eigenvalues: the two extreme ones and the two straddling the origin, in which case shifting can be applied as a form of preconditioning.

Acknowledgments. We thank Aftab Patel for assistance on this paper. We also gratefully acknowledge use of the Blues cluster in the Laboratory Computing

Resource Center at Argonne National Laboratory.

REFERENCES

- [1] L. S. BLACKFORD, J. CHOI, A. CLEARY, E. D’AZEVEDO, J. DEMMEL, I. DHILLON, J. DONGARRA, S. HAMMARLING, G. HENRY, A. PETITET, K. STANLEY, D. WALKER, AND R. C. WHALEY, *ScaLAPACK Users’ Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1997.
- [2] E. R. DAVIDSON, *The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices*, J. Comput. Phys., 17 (1975), pp. 87–94.
- [3] T. A. DAVIS, *Direct Methods for Sparse Linear Systems*, SIAM, 2006.
- [4] T. H. DUNNING JR., *Gaussian basis sets for use in correlated molecular calculations. I. The atoms boron through neon and hydrogen*, The Journal of Chemical Physics, 90 (1989), p. 1007.
- [5] T. ERICSSON AND A. RUHE, *The spectral transformation Lánczos method for the numerical solution of large sparse generalized symmetric eigenvalue problems*, Math. Comp., 35 (1980), pp. 1251–1268.
- [6] H.-R. FANG AND Y. SAAD, *A filtered Lanczos procedure for extreme and interior eigenvalue problems*, SIAM J. Sci. Comput., 34 (2012), pp. A2220–A2246.
- [7] D. R. FOKKEMA, G. L. G. SLEIJPEN, AND H. A. V. DER, *Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils*, SIAM J. Sci. Comput., 20 (1998), pp. 94–125.
- [8] A. FROMMER, T. LIPPERT, B. MEDEKE, AND K. SCHILLING, eds., *Numerical Challenges in Lattice Quantum Chromodynamics*, vol. 15 of Lecture Notes in Computational Science and Engineering, Springer, 2000.
- [9] N. J. HIGHAM, *Functions of Matrices: Theory and Computation*, SIAM, 2008.
- [10] C. S. KENNEY AND A. J. LAUB, *Rational iterative methods for the matrix sign function*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 273–291.
- [11] ———, *The matrix sign function*, IEEE Transactions on Automatic Control, 40 (1995), pp. 1330–1348.
- [12] R. B. LEHOUCQ AND D. C. SORENSEN, *Deflation techniques for an implicitly re-started Arnoldi iteration*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 789–821.
- [13] R. B. LEHOUCQ, D. C. SORENSEN, AND C. YANG, *ARPACK Users’ Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, SIAM, 1998.
- [14] X.-P. LI, R. W. NUNES, AND D. VANDERBILT, *Density-matrix electronic-structure method with linear system-size scaling*, Phys. Rev. B, 47 (1993), pp. 10891–10894.
- [15] X. S. LI, *An overview of SuperLU: Algorithms, implementation, and user interface*, ACM Trans. Math. Softw., 31 (2005), pp. 302–325.
- [16] V. LOTRICH, N. FLOCKE, M. PONTON, A. YAU, A. PERERA, E. DEUMENS, AND R. BARTLETT, *Parallel implementation of electronic structure energy, gradient, and Hessian calculations*, The Journal of Chemical Physics, 128 (2008), p. 194104.
- [17] R. MCWEENY, *Some recent advances in density matrix theory*, Rev. Mod. Phys., 32 (1960), pp. 335–369.
- [18] R. B. MORGAN AND D. S. SCOTT, *Generalizations of Davidson’s method for computing eigenvalues of sparse symmetric matrices*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 817–825.
- [19] H. NEUBERGER, *Exactly massless quarks on the lattice*, Physics Letters B, 417 (1998), pp. 141–144.
- [20] Y. SAAD, *Numerical Methods for Large Eigenvalue Problems, Revised Edition*, SIAM, 2011.
- [21] Y. SAAD, J. CHELIKOWSKY, AND S. SHONTZ, *Numerical methods for electronic structure calculations of materials*, SIAM Review, 52 (2010), pp. 3–54.
- [22] D. C. SORENSEN, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.
- [23] A. SZABO AND N. S. OSTLUND, *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*, Dover, 1989.
- [24] J. VAN DEN ESHOF, A. FROMMER, T. LIPPERT, K. SCHILLING, AND H. A. VAN DER VORST, *Numerical methods for the QCD overlap operator: I. sign-function and error bounds*, Computer Physics Communications, 146 (2002), pp. 203–224.
- [25] S. WANG, X. S. LI, J. XIA, Y. SITU, AND M. V. D. HOOP, *Efficient scalable algorithms for solving linear systems with hierarchically semiseparable structures*. submitted to SIAM J. Sci. Comput., 2012.

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (“Argonne”). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.